Documentation and Tutorial for open_connectedpapers

Introduction

The `open_connectedpapers` project aims to replicate the functionality of a scholarly paper sharing website, specifically targeting Connected Papers. The objective is to crawl data from scholarly articles available on Google Scholar, visualize various attributes of these papers such as citation counts, publication years, and their inter-citation relationships.

Features

1. **Web Scraping:** Utilizes web scraping techniques to gather paper data from Google Scholar.
2. **Data Parsing:** Parses HTML content using BeautifulSoup to extract paper metadata like title, URL, publication year, and citation count.
3. **Citation Graph:** Constructs a citation graph to visualize the relationships between papers.
4. **Visualization:** Employs Pyecharts to visualize the citation graph with nodes representing papers and edges representing citations.
5. **Interactive Interface:** Users can explore the citation graph interactively.

Dependencies

- Python 3.x
- Selenium: For web automation.
- BeautifulSoup: For HTML parsing.
- Pyecharts: For data visualization.
- Matplotlib: For color mapping.

Installation

Ensure you have Python installed on your system. Then, install the required packages using pip:

```
pip install selenium beautifulsoup4 pyecharts matplotlib
```

Usage Guide

1. Setting Up

Before running the code, make sure you have the necessary drivers for Selenium. You can download the appropriate driver for your browser (e.g., Edge) from the Selenium website.

2. Running the Code

Execute the provided Python script `open_connectedpapers.py`:

```
python open_connectedpapers.py
```

3. Understanding the Code

The code consists of several functions to perform different tasks:

- `build_graph()`: Placeholder function to build the citation graph (yet to be implemented).
- `parse_paper_div(paper_div, papers_data)`: Extracts paper information from HTML div elements.
- `get_all_papers_info(paper_url)`: Gathers paper data by crawling Google Scholar.
- `get_paper_data()`: Retrieves paper data, including citation information, and saves it in JSON format.
- `visualize()`: Visualizes the citation graph using Pyecharts.

4. Customization

You can customize the code according to your requirements:

- Adjust web scraping parameters such as URLs, search queries, or pagination settings.
- Modify visualization settings like node size, color scheme, or graph layout.

Conclusion

The `open_connectedpapers` project provides a foundation for exploring scholarly paper citation networks. By leveraging web scraping and visualization techniques, it enables users to analyze and visualize citation patterns, facilitating research and academic exploration.

Disclaimer

Please use this tool responsibly and respect the terms of service of the websites from which data is being scraped. Unauthorized or excessive web scraping may violate these terms and may have legal implications.